CLAIMS

What is claimed is:

1. In a computer, a computer-implemented method for executing an application system call, said application system call involving invoking a kernel thread from a system call stub in a user space of an operating system of said computer, comprising:

saving user space context data in a save state upon entering a kernel space of said operating system from said user space; and

thereafter modifying a return pointer in said save state to an address of a unblock handler call stub in user space instead of an address of said system call stub in said user space, thereby causing said kernel thread to return to said unblock call handler stub instead of returning to said system call stub when said kernel thread completes execution.

2. The computer-implemented method of claim 1 further including registering said address of said unblock call handler stub with a kernel of said operating system prior to said executing said application said system call.

3. The computer-implemented method of claim 2 wherein said registering is performed during application startup.

4. The computer-implemented method of claim 2 wherein said registering is performed during a thread library startup.

5. The computer-implemented method of claim 2 further including registering a data pointer associated with said address of said unblock call handler stub with said kernel.

6. The computer-implemented method of claim 1 further including passing, after said saving, a data pointer associated with said system call stub as an argument to said unblock call handler stub.

7. The computer-implemented method of claim 6 further including passing, after said saving, a return value of said system call and an address of said system call stub as arguments to said unblock call handler stub.

8. The computer-implemented method of claim 7 further including branching from said unblock call handler stub to an unblock call handler routine in said user space.

9. The computer-implemented method of claim 8 further including branching from said unblock call handler routine to said system call stub in said user space when said unblock call handler routine finishes executing without returning to said kernel space.

10. The computer-implemented method of claim 8 wherein said unblock call hander routine includes code for notifying a scheduler in said user space that said kernel thread is unblocked.

11. The computer-implemented method of claim 1 wherein said computer implements a PA-RISC™ architecture.

12. The computer-implemented method of claim 1 wherein said computer implements a PA-IPF™ architecture.

13. The computer-implemented method of claim 1 wherein said computer implements MxN threading.

14. The computer-implemented method of claim 1 wherein said system call involves accessing an I/O device.

15. An article of manufacture comprising a program storage medium having computer readable code embodied therein, said computer readable code being configured for executing an application system call, said application system call involving invoking a kernel thread from a system call stub in a user space of an operating system of said computer, comprising:

code for saving user space context data in a save state upon entering a kernel space of said operating system from said user space; and

code for modifying, after said saving, a return pointer in said save state to an address of a unblock handler call stub in user space instead of an address of said system call stub in said user space, thereby causing said kernel thread to return to said unblock call handler stub instead of returning to said system call stub when said kernel thread completes execution.

16. The article of manufacture of claim 15 further including code for registering said address of said unblock call handler stub with a kernel of said operating system prior to said executing said application said system call.

17. The article of manufacture of claim 16 wherein said registering is performed during one of an application startup and a thread library startup.

18. The article of manufacture of claim 16 further including code for registering a data pointer associated with said address of said unblock call handler stub with said kernel.

19. The article of manufacture of claim 15 further including code for passing, after said saving, at least one of a data pointer associated with said system call stub, a return value of said system call and an address of said system call stub as an argument to said unblock call handler stub.

20. The article of manufacture of claim 15 further including code for branching from said unblock call handler stub to an unblock call handler routine in said user space.

21. The article of manufacture of claim 20 further including code for branching from said unblock call handler routine to said system call stub in said user space when said unblock call handler routine finishes executing without returning to said kernel space.

22. The article of manufacture of claim 22 wherein said unblock call hander routine includes code for notifying a scheduler in said user space that said kernel thread is unblocked.

23. The article of manufacture of claim 22 wherein said computer implements MxN threading.

24. In a computer implementing MxN threading, a computer-implemented method for executing an application system call from a user space of an operating system of said computer, said application system call involving invoking a kernel thread in a kernel space of said operating system from a system call stub in a user space of said operating system, comprising:

saving user space context data in a save state upon entering said kernel space;

thereafter modifying a return pointer in said save state to an address of a unblock handler call stub in user space instead of an address of said system call stub in said user space, thereby causing said kernel thread to return to said unblock call handler stub instead of returning to said system call stub when said kernel thread completes execution, and

returning from said unblock call handler stub in said user space to said system call stub in said user space without entering said kernel space again.

25. The computer-implemented method of claim 24 further including passing, after said saving, a data pointer associated with said system call stub, a return value of said system call, and an address of said system call stub as arguments to said unblock call handler stub.

26. The computer-implemented method of claim 24 further including registering said address of said unblock call handler stub with a kernel of said operating system prior to starting up of an application executing said application system call.

27. The computer-implemented method of claim 24 further including branching from said unblock call handler stub to an unblock call handler routine in said user space prior to branching to said system call stub.

28. The computer-implemented method of claim 27 wherein said unblock call hander routine includes code for notifying a scheduler in said user space that said kernel thread is unblocked.